row 42 may be read. CPU 22 may use this value to calculate the last valid data byte it received from last data row 40.

[0034] In a further embodiment of the present invention, DMA 16/26 may calculate the amount of data received.

[0035] FIFO 18 may provide flow control for data transmission and receipt to DMAs 16 and 26. DMAs 16 and 26 may start and stop data flow in response to the various signals described. Thus, as data may not be subject to overflow/underflow conditions, it may be possible to have only 2 - 4 rows.

[0036] Reference is now made to Figs. 4A and 4B, which are flow chart diagrams of the functionality implemented in chip 8 (of Fig. 1), in accordance with an embodiment of the present invention. Other implementations are possible and are within the scope of the present invention. Fig. 4A shows the steps performed on the transmission side, whereas Fig. 4B depicts the steps of the receive side.

[0037] Hereinbelow reference is made to CPU 12, RAM 14, DMA 16, system data bus 17, and FIFO 18. Equivalently, CPU 22, RAM 24, DMA 26, system data bus 27, and FIFO 28 could be used.

[0038] The data transmission process will be described first. CPU 12 may instruct DMA 16 to transmit a given amount of data found in RAM 14 at a given location (step 101). If FIFO 18 is full it may transmit a FIFO full signal to DMA 16 (step 103), which may cause DMA 16 to wait until data is read from FIFO 18, thus freeing space for DMA 16 to write to.

[0039] If no FIFO full signal is received, DMA 16 may send a WR signal to control 32 of FIFO 18. DMA 16 may write a row of data into FIFO 18 (step 105). Additionally, an appropriate value, for example a "0" indicating a row of data, may

be entered in EOP flags 44. A check may be performed to see if all the data has been written (step 107). If not, steps 103, 105, and 107 may be repeated until all data has been written to FIFO 18. As mentioned previously, the last row of data, last data row 40, may contain invalid data (garbage) if the data is not aligned to fill the row.

[0040] Once all data is written, an additional row may be written into data block 34, delimiter row 42 (step 109). This row may, for example, contain the number of the last byte containing actual data in the previous row (last data row 40). In another embodiment, for example, delimiter row 42 may contain the length of the data. At generally the same time, DMA 16 may send an EOP_in signal to FIFO 18 (step 109) causing an appropriate value, for example a value indicating end of data (such as a 1), to be written into EOP flags 44.

[0041] DMA 16 may further send a signal to CPU 12 indicating that the data has been transmitted (step 111). At this point DMA 16 may be finished with the current data transmission. It is available if another data transmission is required, possibly from another memory location in RAM 14 (optional return to step 101).

[0042] Data receipt will now be described. CPU 12 may instruct DMA 16 to receive data and to write it to a given location in RAM 14 (step 201). DMA 16 may check if FIFO 18 is empty (step 203), for example, if it received a FIFO empty signal. If FIFO 18 is empty, DMA 16 may wait until there is data. If FIFO 18 contains data, DMA 16 may send a RD signal to FIFO 18 and may begin reading the data one row at a time (step 205).

[0043] After each data row is read, a check may be made (step 207) to determine whether an EOP_out signal has been sent by FIFO 18. If EOP flags 44 indicates the

end of data, for example by containing a 1, the EOP_out signal may be sent. If an EOP_out signal is not received, then steps 203, 205, and 207 may be performed until all data is read.

[0044] Upon receipt of the EOP_out signal, DMA 16 may store the delimiter and the number of received data rows (step 209) and may further send an interrupt signal to CPU 12 informing it of the receipt of the data transmission (step 210). Upon receipt of the interrupt, CPU 22 may read the delimiter and the number of received data rows and calculate how much data was received.

[0045] The data may be processed by CPU 12 (step 211). Processing may include calculating the amount of data received. The number of full rows of data may be multiplied by the number of bytes in a row to calculate an "initial total bytes" value. The number of bytes of actual data in the last data row, which is given in end delimiter row, may be added to initial total bytes producing "total bytes". Total bytes may be the number of bytes of data that were received. Other data processing may also be done by CPU 12.

[0046] CPU 12 may re-instruct DMA 16 to await data reception but may change the location in RAM 14 to write to (return to step 201). This may be done after step 210 or step 211.

[0047] There may be systems that use asynchronous clocks to control CPU activities. Asynchronous clocks may have different clock speeds or different phases, which may allow different rates of transmission to be set by different clocks. In a system using asynchronous clocks, synchronization between different CPUs which must share information may be complicated. Since the present invention may comprise FIFOs, these same FIFOs may also be used for data synchronization.